MACHINE LEARNING MODEL FOR PREDICTION OF CALORIES BURNT DURING EXERCISE

REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF

BACHELOR OF TECHNOLOGY IN INFORMATION TECHNOLOGY

BY

Arjun Gautam Baruah

220102001

Ashmit karmakar

220102008

UNDER THE GUIDANCE

OF

Prof. Shikhar Kumar Sarma

Department of Information Technology

Gauhati University

DEPARTMENT OF INFORMATION TECHNOLOGY
GAUHATI UNIVERSITY
GUWAHATI, INDIA

MAY, 2025

GAUHATI UNIVERSITY

DEPARTMENT OF INFORMATION TECHNOLOGY

Gopinath Bordoloi Nagar, Jalukbari, Guwahati-781014

DECLARATION

We, *Arjun Gautam Baruah* and *Ashmit Karmakar*, Roll No: *220102001* and *220102008* respectively, B.Tech. students of the department of Information Technology, Gauhati University hereby declare that we have compiled this report reflecting all our works during the semester long full time project as part of our BTech curriculum.

We declare that we have included the descriptions etc. of our project work, and nothing has been copied/replicated from other's work. The facts, figures, analysis, results, claims etc. depicted in our thesis are all related to our full time project work.

We also declare that the same report or any substantial portion of this report has not been submitted anywhere else as part of any requirements for any degree/diploma etc.

Arjun Gautam Baruah (220102001)
Branch: Information Technology
Date: 30/05/2025

Ashmit Karmakar (220102008)
Branch: Information Technology
Date: 30/05/2025

GAUHATI UNIVERSITY

DEPARTMENT OF INFORMATION TECHNOLOGY

Gopinath Bordoloi Nagar, Jalukbari, Guwahati-781014

Date: 30/05/2025

CERTIFICATE

This is to certify that *Arjun Gautam Baruah* and *Ashmit Karmakar* bearing Roll No: *220102001* and *220102008* have carried out the project work *ML Model for Prediction of Calories Burnt during Exercise* under my supervision and has compiled this report reflecting the candidates' work in the semester long project. The candidates did this project full time during the whole semester under my supervision, and the analysis, results, claims etc. are all related to their studies and works during the semester.

I recommend submission of this project report as a part for partial fulfillment of the requirements for the degree of Bachelor of Technology in Information Technology of Gauhati University.

Prof. Shikhar Kumar Sarma
Head of the Department
Department of Information Technology
Gauhati University

GAUHATI UNIVERSITY

DEPARTMENT OF INFORMATION TECHNOLOGY

Gopinath Bordoloi Nagar, Jalukbari, Guwahati-781014

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to all those who supported us throughout the work of our project work. We are thankful for their aspiring guidance, valuable time, invaluably constructive and friendly advice during the entire period.

We convey our special thanks to Prof. Shikhar Kumar Sarma for their constant support and guidance at the Department of Information Technology, Gauhati University, Guwahati during our project generation.

Thanking you all.

Sincerely,
Arjun Gautam Baruah
Ashmit Karmakar
B.Tech., 6th Semester, Dept. of Information Technology,
Gauhati University,
GUWAHATI

GAUHATI UNIVERSITY

DEPARTMENT OF INFORMATION TECHNOLOGY

Gopinath Bordoloi Nagar, Jalukbari, Guwahati-781014

ABSTRACT

This project presents a machine learning-based system for *Prediction of Calories Brunt during Exercise,* aimed at helping users understand the energy expenditure during physical activity. The model uses an enriched dataset that combines physical attributes such as age, gender, weight, body temperature, heart rate, and exercise duration with additional engineered features like calories consumed before workout, seasonal variations, and customized MET values derived from exercise types and intensity.

The system is developed using the XGBoost Regressor, a powerful and efficient gradient boosting algorithm, trained on this customized dataset. Advanced feature engineering techniques are applied to improve model accuracy by quantifying the individual impact of each parameter on calorie burning.

An interactive user input section allows users to enter their details and receive an estimated number of calories burnt along with key metrics. This project can serve as a useful tool for fitness enthusiasts, health professionals, and anyone seeking to manage their health through data-driven insights.

TABLE OF CONTENTS

OBJECTIVE

Problem Definition:

Creating a ML Model to predict calories burnt during exercise, using an existing dataset and then adding riders to the dataset such as season, exercises done during the workout, calories consumed before workout, etc.

The main objective of this project is to develop an intelligent and efficient system that can predict the number of calories burnt by a person based on their physical attributes and exercise activity. This model is designed to assist users in monitoring their fitness and managing their energy expenditure more accurately.

To achieve this, the project uses machine learning techniques, specifically the XGBoost Regressor, to train a model on enriched fitness data. The system incorporates **feature engineering** to enhance prediction accuracy by considering factors such as MET values, heart rate, body temperature, season, and calories consumed before exercise.

This project aims to create a user-interactive platform where users can input their details and receive real-time feedback, ultimately promoting better awareness and control over personal health and fitness goals.

INTRODUCTION

The “**Machine Learning Model for Prediction of Calories Burnt During Exercise**” is a machine learning-based project designed to estimate the number of calories a person burns during physical activity. With the rise in health consciousness and fitness tracking, this project aims to provide an intelligent, personalized tool that factors in various physiological and activity-based parameters to predict calorie expenditure accurately.

The system uses real-world data and incorporates important user features such as gender, age, weight, body temperature, heart rate, duration of activity, and even seasonal variations. Additionally, the model considers the type of exercises performed and calculates an average **MET (Metabolic Equivalent of Task)** value based on heart rate intensity levels. These inputs are further enhanced through feature engineering to represent their relative impact on calorie burn.

MET stands for **Metabolic Equivalent of Task**, a unit that estimates energy expenditure during physical activities:

1 MET = energy cost of sitting quietly (~1 kcal/kg/hour)

So if an activity is 8 METs, it burns 8 times more energy than resting.

1. A 60-year-old burns fewer calories than a 20-year-old doing the same MET-8 activity.
2. In cold weather, body burns more energy to maintain temperature.
3. A female might burn fewer calories than a male due to muscle mass differences.
4. Someone who just ate will burn energy differently due to digestion.

By employing the **XGBoost Regression algorithm**, known for its speed and accuracy, the model learns from the data and delivers precise predictions. A user-friendly interface allows individuals to input their data and receive instant feedback on their estimated calorie burn. This project showcases how artificial intelligence can be used in the fitness and healthcare domain to assist individuals in monitoring and optimizing their workout routines effectively.

The novelty of this project is the feature engineering, we wanted to consider the importance of the somewhat less important features such as Age, Gender, Season, Body Temperature, Calories Consumed before workout throughout the day.

How this subtle yet important features influence calories burnt during exercise.

These features have been assigned weights based on research on how important the feature is compared to the other.

In the case of label encoding seasons,

For eg: We have ordered the seasons such that during label encoding: "winter" which is the most favourable season to burn calories is assigned 4 and summer assigned 0 accordingly, which is taken into consideration when we have feature engineered a Custom MET Value assigning weights based on research to each feature.

Also the use of XGBoostRegressor from XGBoost which is a type of boosting in ensemble learning where more than one learner in which one weak learner learns from the error of the previous learner.

The learner specifically is a decision tree.

The model might seem more formula based than learning based but to signify the importance of the less important features.

Perhaps, the XGBoost model might learn subtle patterns, and non-linear relationships among the features which might influence the model’s prediction, also because the dataset has 15000 datapoints, it is vast enough to learn patterns.

Background and Literature Review

Background of Calorie Burn Prediction

- Traditional calorie burn calculations often use fixed formulas like METs × weight × duration, which ignore individual variations.

- Machine Learning (ML) enables more personalized predictions by learning patterns from real-world data.

- Modern fitness applications increasingly rely on ML models to estimate energy expenditure based on multiple parameters.

- Features like heart rate, gender, age, body temperature, and exercise type greatly influence calorie burn but are often oversimplified in standard methods.

- ML algorithms, especially XGBoost, handle complex, non-linear relationships between inputs and output (calories burnt).

- Incorporating feature engineering (e.g., seasonal effects, calories consumed before exercise) further improves prediction accuracy.

- ML thus provides a smarter, data-driven solution for personalized health and fitness tracking.

Literature Review of Machine Learning Calories burnt Predictions

Machine learning algorithms have gained widespread use in recent years to predict calorie burn during physical activity. These studies often collect physical activity data and other relevant variables such as heart rate, age, and gender from fitness trackers, mobile applications, and wearable devices. This section provides an overview of some of the critical studies in this area.

Sathiya T et al. discussed to predict user's calorie and applied CNN model to classify food items from the input image. They also used image processing techniques such as deep learning model and their model provide 91.65% accuracy in predicting user's calorie from input image.

Sona P Vinoy illustrates to predict calorie burn during the workout et al. used machine learning algorithms such as XGBboost regressor and Linear regression models to and out calorie burnt in physical activities. Their mean absolute error value is almost 2.71 in XGB regressor and 8.31 for linear regression. They used 7 attributes such as age, height, weight, duration, heart_rate, body_temp and calorie. Their dataset was in 15000 CSV with 7 attributes. They did not mention their model accuracy.

Suvarna Shreyas Ratnakar et al. discussed how to predict calories burnt from physical activities. They used the XGB boost Machine learning algorithm to predict it including 15,000 raw dataset and their mean absolute error value is 2.7 and model accuracy is not mentioned. Rachit Kumar Singh et al. illustrated their method to predict calorie burn using machine learning techniques. In their work, logistic regression, linear regression and lasso regression models were used but they didn't mention about mean error absolute value, dataset and model accuracy.

Marte Nipas et al. discussed how to predict burned calories using a supervised learning algorithm. They used a Random forest algorithm and gained 95.77% model accuracy. They also used the iterative method to and out the appropriate output from an input. Their work is almost better than other recent work.

Gunasheela B L et al. discussed their techniques to predict calorie from input images. They used some digital image processing techniques such as image acquisition, RGB conversion, feature extraction and image enhancement so on. They segmented input images and used techniques and then combined segmented images, finally calorie predicted.

KR Westerterp et al. discussed how to determine energy expenditure by body size and body compositions and food intake and physical activity. He used body size and body compositions and some statistical techniques to evaluate calorie expenditure.In summary, these studies demonstrate the potential for machine learning algorithms to predict energy expenditure accurately during physical activity. However, there is still a need for models that can accurately predict energy expenditure across various physical activities and individuals

METHODOLOGY

The project begins with importing essential libraries for data manipulation, visualization, and machine learning. Next, the dataset containing user and exercise-related information is loaded for analysis.

To improve the model's accuracy, feature engineering is performed by deriving new attributes such as MET values, which quantify the intensity of different exercises. Categorical data like gender and exercise type are encoded into numerical values to enable model compatibility. The dataset is then divided into training and testing subsets to ensure unbiased model evaluation.

For the implementation, an XGBoost regression model is trained using the prepared data, leveraging its efficiency in handling complex, non-linear relationships. The trained model's performance is evaluated using appropriate regression metrics to verify its accuracy.

Visualizations of feature importance provide insights into which factors most influence calorie burn. Finally, a user interface collects real-time input, processes it into the model's expected format, and displays the predicted calories burnt, allowing dynamic and interactive predictions.

DATA PREPARATION AND PROCESSING

1. Importing Required Libraries

At the beginning of the project, we imported all the essential Python libraries that were required for building the Calories Burnt Prediction system. These libraries provide various tools that make data processing, modeling, and evaluation much easier.

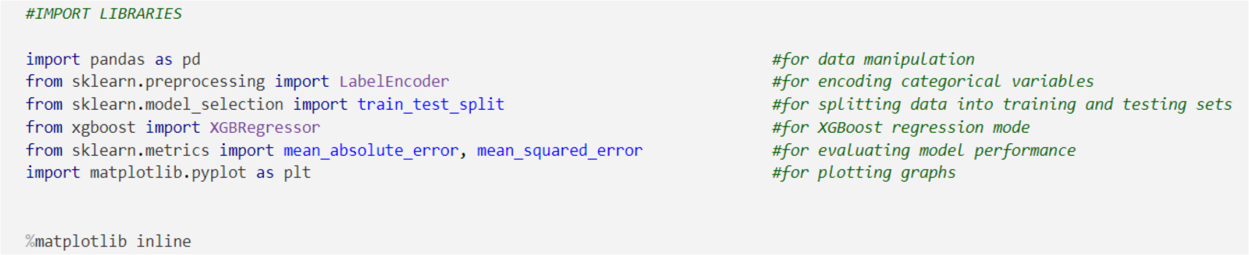
First, we used the pandas library which is commonly used for handling datasets in the form of tables (called DataFrames). It allows us to load, explore, and manipulate data efficiently. Next, we imported LabelEncoder from the sklearn.preprocessing module. This is used to convert categorical text data (like gender: male or female) into numerical form, which is necessary because machine learning models can only understand numbers.

The train_test_split function from sklearn.model_selection is used to divide our dataset into two parts — one for training the model and the other for testing it. This helps in checking whether the model performs well on unseen data.

For building the prediction model, we used the XGBRegressor from the XGBoost library. XGBoost is a powerful and efficient machine learning algorithm known for its high accuracy and performance.

We also imported evaluation metrics such as mean_absolute_error, mean_squared_error, and r2_score to measure how well our model is predicting the calories burnt.

Lastly, we used matplotlib.pyplot to create visualizations, and %matplotlib inline to make sure plots appear inside the notebook itself.



```
#IMPORT LIBRARIES

import pandas as pd                                                              #for data manipulation
from sklearn.preprocessing import LabelEncoder                                   #for encoding categorical variables
from sklearn.model_selection import train_test_split                             #for splitting data into training and testing sets
from xgboost import XGBRegressor                                                 #for XGBoost regression mode
from sklearn.metrics import mean_absolute_error, mean_squared_error              #for evaluating model performance
import matplotlib.pyplot as plt                                                  #for plotting graphs


%matplotlib inline
```

2. Loading the Datasets

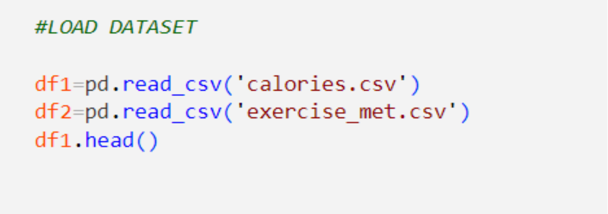
In this step, we loaded the two datasets required for the Calories Burnt Prediction system. The data was provided in csv (Comma-Separated Values) format, which is a common file type for storing tabular data.

The first file, calories.csv, contains the main data related to the individuals and the calories they have burnt. It includes features like gender, age, height, weight, duration of exercise, and the number of calories burnt. This dataset was loaded into a DataFrame named df1.

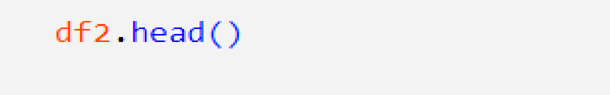
The second file, exercise_met.csv, contains the MET (Metabolic Equivalent of Task) values for different types of exercises. MET is a unit used to estimate the energy used by the body during physical activities. This file was loaded into a DataFrame called df2.

After loading both datasets using the read_csv() function from the pandas library, we used the .head() function to view the first five rows of each dataset. This step is important to quickly inspect the structure of the data and understand what kind of columns and values are present. It also helps to check if the data was loaded correctly without any errors.

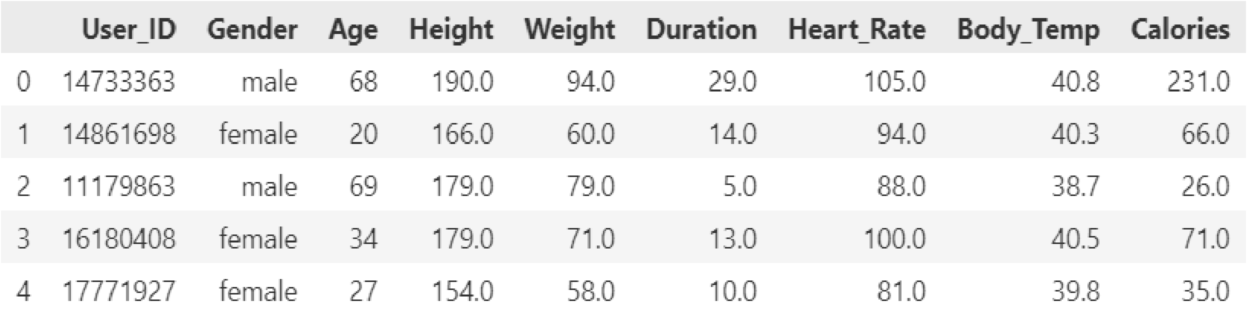
This data loading step is the foundation of the project as all further processing depends on this raw data.



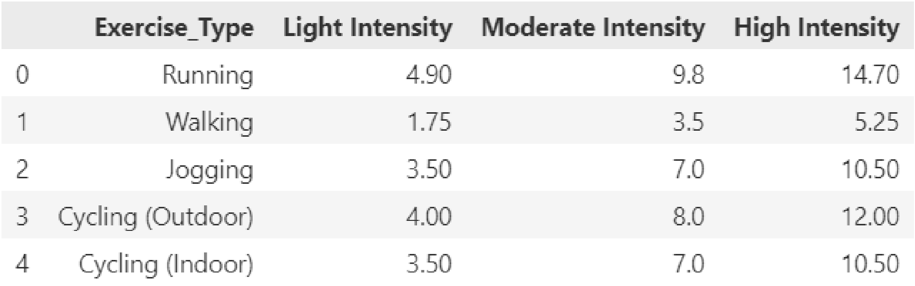
```
#LOAD DATASET

df1=pd.read_csv('calories.csv')
df2=pd.read_csv('exercise_met.csv')
df1.head()
```



```
df2.head()
```



	User_ID	Gender	Age	Height	Weight	Duration	Heart_Rate	Body_Temp	Calories
0	14733363	male	68	190.0	94.0	29.0	105.0	40.8	231.0
1	14861698	female	20	166.0	60.0	14.0	94.0	40.3	66.0
2	11179863	male	69	179.0	79.0	5.0	88.0	38.7	26.0
3	16180408	female	34	179.0	71.0	13.0	100.0	40.5	71.0
4	17771927	female	27	154.0	58.0	10.0	81.0	39.8	35.0



	Exercise_Type	Light Intensity	Moderate Intensity	High Intensity
0	Running	4.90	9.8	14.70
1	Walking	1.75	3.5	5.25
2	Jogging	3.50	7.0	10.50
3	Cycling (Outdoor)	4.00	8.0	12.00
4	Cycling (Indoor)	3.50	7.0	10.50

3. Feature Engineering

Feature engineering is one of the most important steps in any machine learning project. In this section, we created a few new columns (or features) in the dataset to make the model more informative and improve its prediction ability.

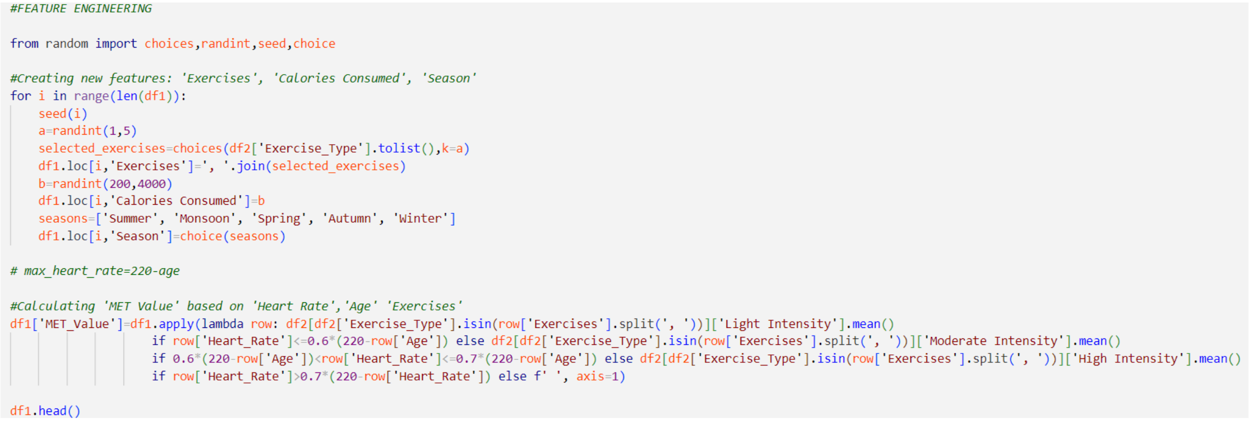
We began by importing random functions to help generate new data features. Using these, we added three new columns to the main dataset: Exercises, Calories Consumed, and Season.

For each person (each row in the dataset), we selected a random number of exercises (between 1 and 5) from the exercise list in the exercise_met.csv file. These selected exercise names were joined as a single string and stored in the new Exercises column. Then we randomly assigned a number between 200 and 4000 to represent the calories the person consumed, and this was stored in the Calories Consumed column. Finally, we assigned a random season (like Summer, Winter, etc.) to each record in the Season column to reflect possible seasonal effects on calorie burn.

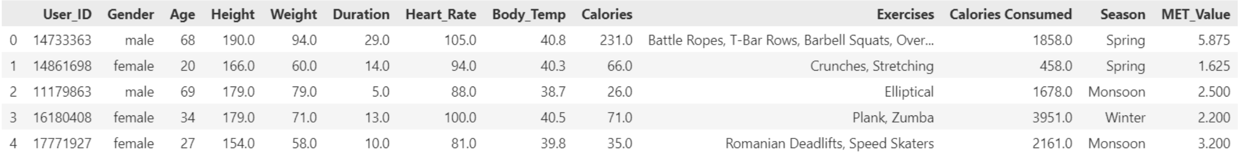
The intensity of a workout is determined by the maximum heart rate of the person which is dependent on the person’s age.

Next, we calculated a new column called MET_Value. This value depends on the person's heart rate and the type of exercises they did. If their heart rate was low, we took the average of “Light Intensity” MET values. For moderate and high heart rates, we used “Moderate Intensity” or “High Intensity” values accordingly. This gave a more realistic representation of energy spent during the workout.

This step enriched the dataset and added more useful information for the model to learn from.



```
#FEATURE ENGINEERING

from random import choices,randint,seed,choice

#Creating new features: 'Exercises', 'Calories Consumed', 'Season'
for i in range(len(df1)):
    seed(i)
    a=randint(1,5)
    selected_exercises=choices(df2['Exercise_Type'].tolist(),k=a)
    df1.loc[i,'Exercises']=', '.join(selected_exercises)
    b=randint(200,4000)
    df1.loc[i,'Calories Consumed']=b
    seasons=['Summer', 'Monsoon', 'Spring', 'Autumn', 'Winter']
    df1.loc[i,'Season']=choice(seasons)

# max_heart_rate=220-age

#Calculating 'MET Value' based on 'Heart Rate','Age' 'Exercises'
df1['MET_Value']=df1.apply(lambda row: df2[df2['Exercise_Type'].isin(row['Exercises'].split(', '))]['Light Intensity'].mean()
                    if row['Heart_Rate']<=0.6*(220-row['Age']) else df2[df2['Exercise_Type'].isin(row['Exercises'].split(', '))]['Moderate Intensity'].mean()
                    if 0.6*(220-row['Age'])<row['Heart_Rate']<=0.7*(220-row['Age']) else df2[df2['Exercise_Type'].isin(row['Exercises'].split(', '))]['High Intensity'].mean()
                    if row['Heart_Rate']>0.7*(220-row['Heart_Rate']) else f' ', axis=1)

df1.head()
```



	User_ID	Gender	Age	Height	Weight	Duration	Heart_Rate	Body_Temp	Calories	Exercises	Calories Consumed	Season	MET_Value
0	14733363	male	68	190.0	94.0	29.0	105.0	40.8	231.0	Battle Ropes, T-Bar Rows, Barbell Squats, Over...	1858.0	Spring	5.875
1	14861698	female	20	166.0	60.0	14.0	94.0	40.3	66.0	Crunches, Stretching	458.0	Spring	1.625
2	11179863	male	69	179.0	79.0	5.0	88.0	38.7	26.0	Elliptical	1678.0	Monsoon	2.500
3	16180408	female	34	179.0	71.0	13.0	100.0	40.5	71.0	Plank, Zumba	3951.0	Winter	2.200
4	17771927	female	27	154.0	58.0	10.0	81.0	39.8	35.0	Romanian Deadlifts, Speed Skaters	2161.0	Monsoon	3.200

4. Encoding and Feature Impact Engineering

In this step, we performed encoding and added calculated impact scores to tune the importance of features that affect calories burnt. This is a crucial step to help the model understand how different factors contribute to the outcome.

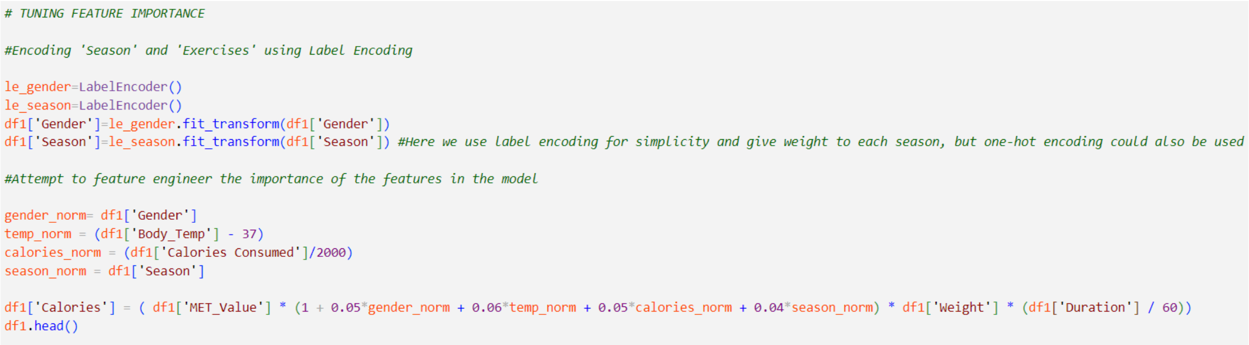
Since machine learning models work only with numbers, we used Label Encoding to convert categorical values like 'Gender' and 'Season' into numeric form. we chose label encoding over one-hot encoding for simplicity, especially because XGBoost can handle encoded categorical features well.

Next, we will normalize the some of the somewhat minor features that might influence each other These were normalized accordingly:

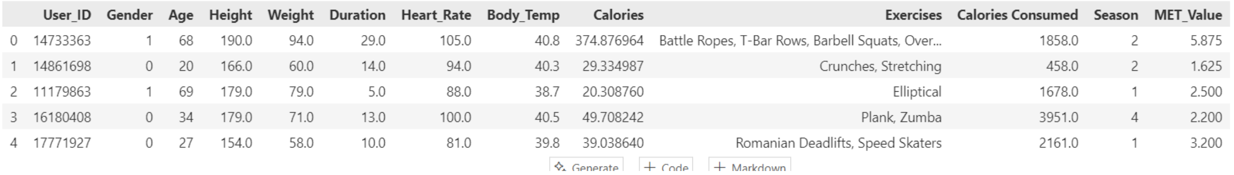
- A slight weight was given based on gender (as males and females may have different energy consumption patterns).
- Higher body temperature may suggest more activity, so a factor was calculated based on deviation from 37°C.
- Calories consumed before workout very slightly and indirectly influence the calories burnt.

- Environmental factors can influence workouts (e.g., winter means more calories burnt vs summer means less calories burnt), so season was included.

Using all of these, we recalculated the *Calories* column with a formula based on MET value, total weight, duration, and these factors.



```
# TUNING FEATURE IMPORTANCE

#Encoding 'Season' and 'Exercises' using Label Encoding

le_gender=LabelEncoder()
le_season=LabelEncoder()
df1['Gender']=le_gender.fit_transform(df1['Gender'])
df1['Season']=le_season.fit_transform(df1['Season']) #Here we use label encoding for simplicity and give weight to each season, but one-hot encoding could also be used

#Attempt to feature engineer the importance of the features in the model

gender_norm= df1['Gender']
temp_norm = (df1['Body_Temp'] - 37)
calories_norm = (df1['Calories Consumed']/2000)
season_norm = df1['Season']

df1['Calories'] = ( df1['MET_Value'] * (1 + 0.05*gender_norm + 0.06*temp_norm + 0.05*calories_norm + 0.04*season_norm) * df1['Weight'] * (df1['Duration'] / 60))
df1.head()
```



	User_ID	Gender	Age	Height	Weight	Duration	Heart_Rate	Body_Temp	Calories	Exercises	Calories Consumed	Season	MET_Value
0	14733363	1	68	190.0	94.0	29.0	105.0	40.8	374.876964	Battle Ropes, T-Bar Rows, Barbell Squats, Over...	1858.0	2	5.875
1	14861698	0	20	166.0	60.0	14.0	94.0	40.3	29.334987	Crunches, Stretching	458.0	2	1.625
2	11179863	1	69	179.0	79.0	5.0	88.0	38.7	20.308760	Elliptical	1678.0	1	2.500
3	16180408	0	34	179.0	71.0	13.0	100.0	40.5	49.708242	Plank, Zumba	3951.0	4	2.200
4	17771927	0	27	154.0	58.0	10.0	81.0	39.8	39.038640	Romanian Deadlifts, Speed Skaters	2161.0	1	3.200

5. Splitting the Dataset into Training and Testing Sets

After preparing the data with all the necessary features and target values, the next important step was to divide the data into two parts: one for training the machine learning model and the other for testing how well the model performs on new, unseen data.

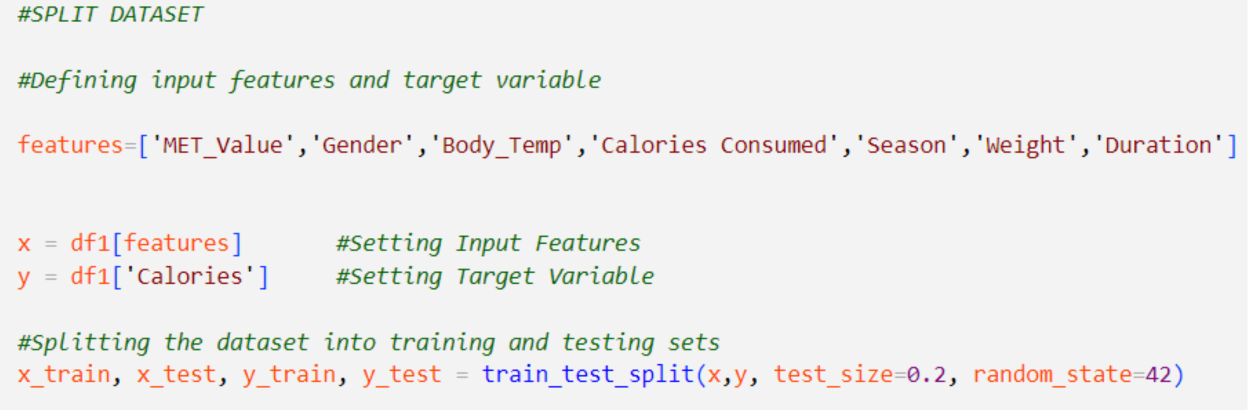
First, we defined the input features (x) that the model will use to predict calories burnt. These features include the MET value, various impact scores like age and gender impact, the person's weight, and the duration of exercise.

The target variable (y) is the actual number of calories burnt, which the model needs to learn to predict.

Using the *train_test_split* function from scikit-learn, we randomly divided the dataset into training and testing sets. we allocated 80% of the data for training the model and kept 20% aside for testing. This split helps to evaluate if the model can generalize well to new data it has not seen before.

The *random_state* parameter ensures that the split is reproducible — the same data division will happen every time the code runs.

Splitting data like this is crucial to avoid overfitting, where a model performs well on training data but poorly on new data.



```
#SPLIT DATASET

#Defining input features and target variable

features=['MET_Value','Gender','Body_Temp','Calories Consumed','Season','Weight','Duration']


x = df1[features]       #Setting Input Features
y = df1['Calories']     #Setting Target Variable

#Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.2, random_state=42)
```

IMPLEMENTATION AND EXPERIMENTATION

1. Training the XGBoost Regression Model

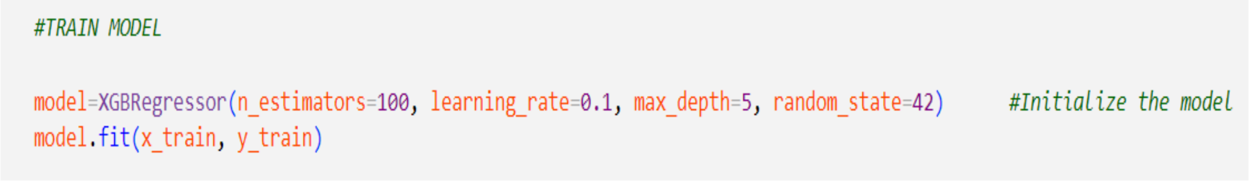
Once the dataset was ready and split into training and testing parts, the next step was to create and train the machine learning model.

In this project, we used the XGBoost Regressor, a powerful and widely used machine learning algorithm especially designed for regression problems like predicting calories burnt. XGBoost is known for its speed, accuracy, and ability to handle complex data patterns.

The model was initialized with some important parameters:

- *n_estimators=100*: This means the model will build 100 trees sequentially to improve predictions.
- *learning_rate=0.1*: This controls how much the model learns at each step; a moderate value helps balance learning speed and accuracy.
- *max_depth=5*: This limits how deep each tree can grow, preventing the model from becoming too complex and overfitting.
- *random_state=42*: Ensures that results are reproducible every time the code is run.

After setting up the model, we trained it using *the .fit()* method on the training dataset (*x_train* and *y_train*). This process allows the model to learn the relationship between the input features and the calories burnt.



```
#TRAIN MODEL

model=XGBRegressor(n_estimators=100, learning_rate=0.1, max_depth=5, random_state=42)      #Initialize the model
model.fit(x_train, y_train)
```

2. Making Predictions and Evaluating the Model

After training the XGBoost regression model, the next step was to test how well it can predict calories burnt on new data that it has never seen before.

Using the *.predict()* method, we generated predictions for the test dataset (*x_test*). These predicted values (*y_pred*) represent the model's estimated calories burnt for each test sample.

To measure the model's accuracy, we used two common evaluation metrics:

- Mean Absolute Error (MAE): This calculates the average absolute difference between the actual calories burnt (*y_test*) and the predicted values *(y_pred*). Lower MAE indicates better performance because predictions are closer to the true values.

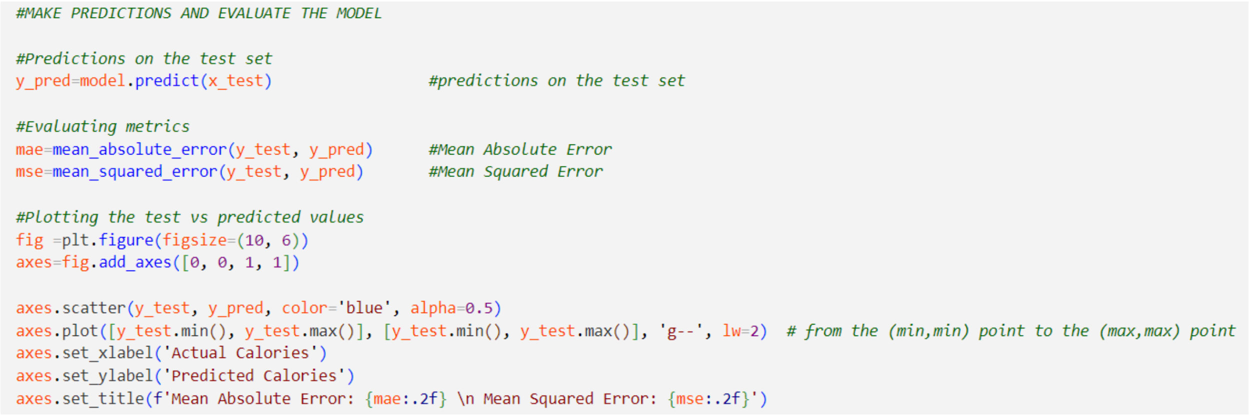
$$MAE = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}|$$

- Mean Squared Error (MSE): This measures the average squared difference, penalizing larger errors more heavily. A smaller MSE also means better accuracy.

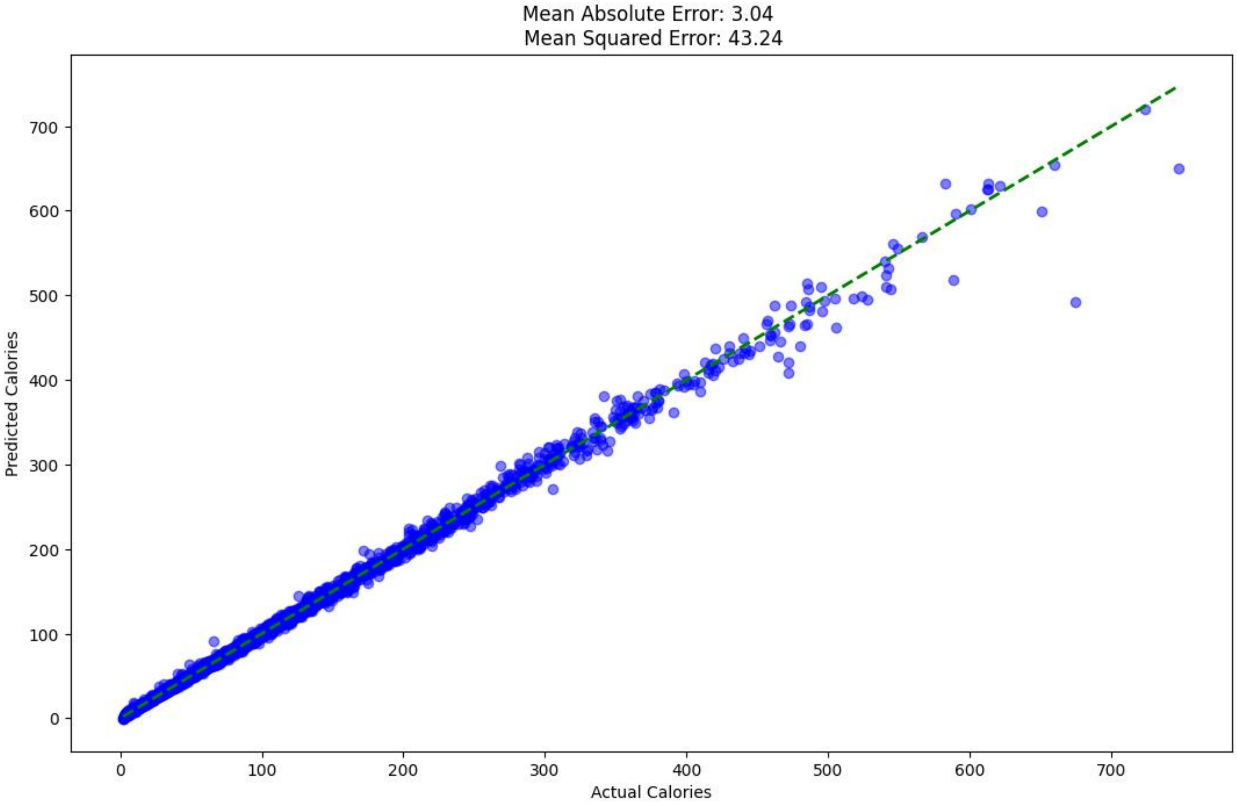
$$MSE = \frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y})^2$$

To visualize the results, we created a scatter plot comparing the actual calories burnt (*y_test*) versus the predicted calories (*y_pred*). The green dashed line shows the ideal prediction (where actual equals predicted). The closer the points are to this line, the better the model's performance.

This evaluation helps to understand the model's accuracy and reliability in predicting calories burnt, which is crucial for its practical use.



```
#MAKE PREDICTIONS AND EVALUATE THE MODEL

#Predictions on the test set
y_pred=model.predict(x_test)                 #predictions on the test set

#Evaluating metrics
mae=mean_absolute_error(y_test, y_pred)      #Mean Absolute Error
mse=mean_squared_error(y_test, y_pred)       #Mean Squared Error

#Plotting the test vs predicted values
fig =plt.figure(figsize=(10, 6))
axes=fig.add_axes([0, 0, 1, 1])

axes.scatter(y_test, y_pred, color='blue', alpha=0.5)
axes.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'g--', lw=2)  # from the (min,min) point to the (max,max) point
axes.set_xlabel('Actual Calories')
axes.set_ylabel('Predicted Calories')
axes.set_title(f'Mean Absolute Error: {mae:.2f} \n Mean Squared Error: {mse:.2f}')
```



3. Plotting Feature Importance

Understanding which features influence the model's predictions the most is crucial for interpreting and improving the model. To achieve this, we extracted the feature importance values from the trained XGBoost model.

This line retrieves the importance scores assigned by the model to each input feature. These scores reflect how much each feature contributes to predicting calories burnt.

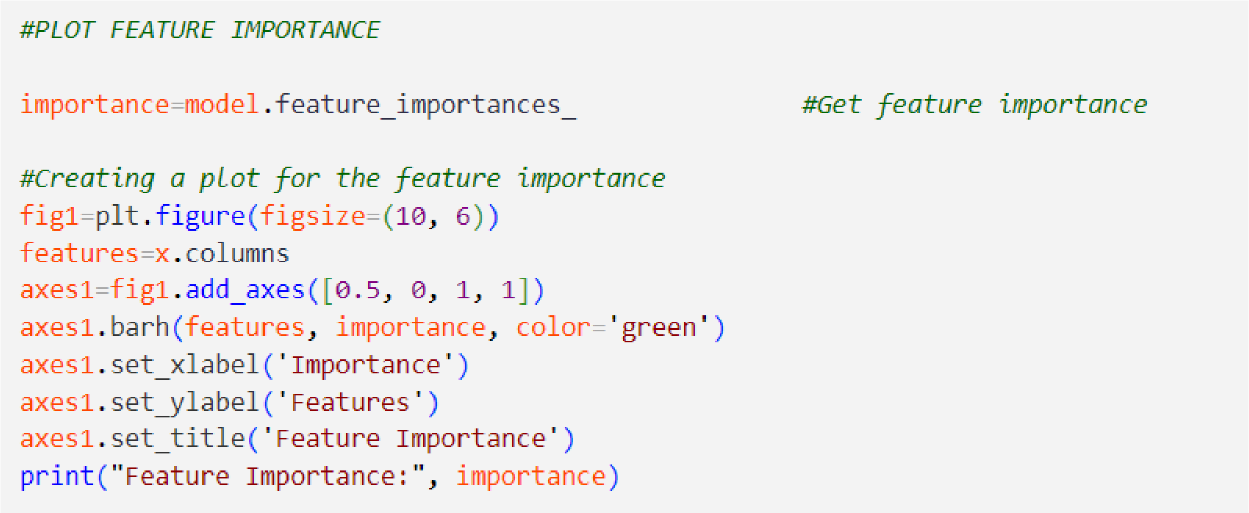
To visualize these scores, we created a horizontal bar plot using Matplotlib:

- The y-axis lists the feature names, such as *MET_Value, Gender, Duration* etc.
- The x-axis shows the importance scores.
- Each bar's length corresponds to the relative importance of that feature.

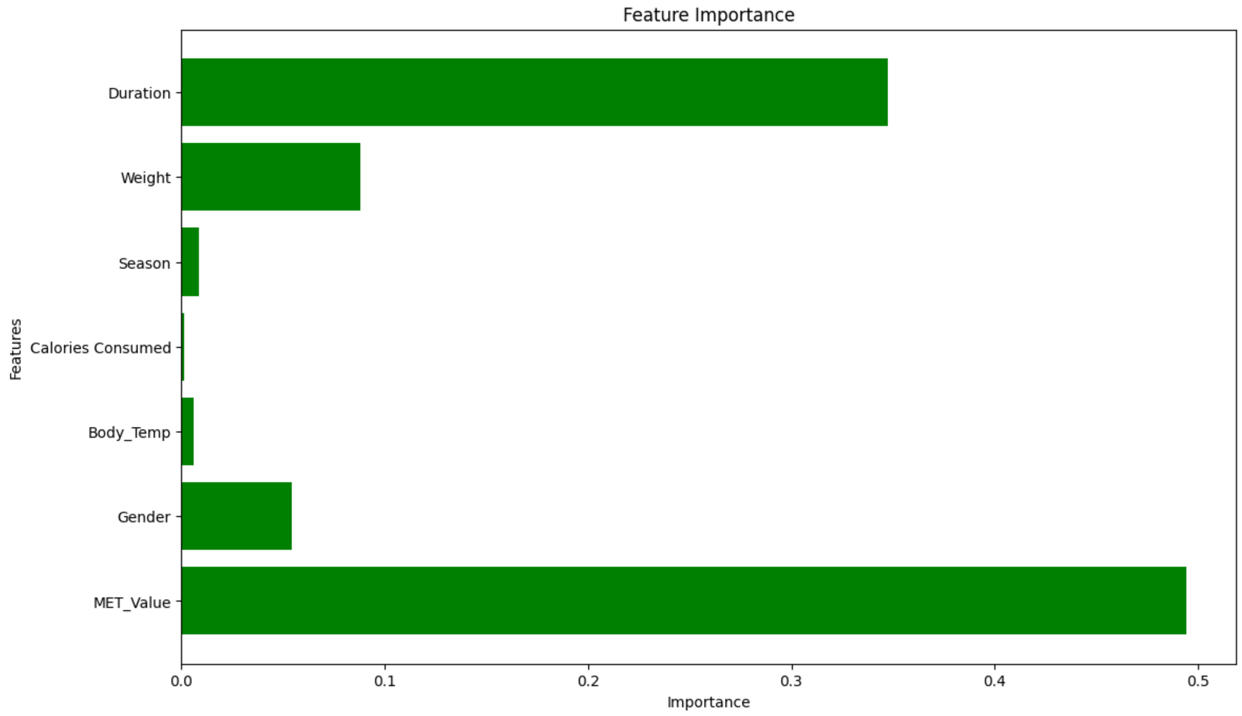
This plot helps identify which features have the greatest impact on calorie prediction. For example, if *MET_Value* has the highest score, it means the model relies heavily on this feature. Features with low importance contribute less to the predictions.

By analyzing this plot, we can better understand the model's decision-making and possibly improve it by focusing on the most influential features.

The printed importance array provides the exact numerical values for reference.



```
#PLOT FEATURE IMPORTANCE

importance=model.feature_importances_                #Get feature importance

#Creating a plot for the feature importance
fig1=plt.figure(figsize=(10, 6))
features=x.columns
axes1=fig1.add_axes([0.5, 0, 1, 1])
axes1.barh(features, importance, color='green')
axes1.set_xlabel('Importance')
axes1.set_ylabel('Features')
axes1.set_title('Feature Importance')
print("Feature Importance:", importance)
```



4. Taking User Input and Preparing Data for Prediction

To make the calorie burn prediction personalized and interactive, we included a section where users can enter their own information. This allows the model to estimate calories burnt based on individual characteristics and exercise details.

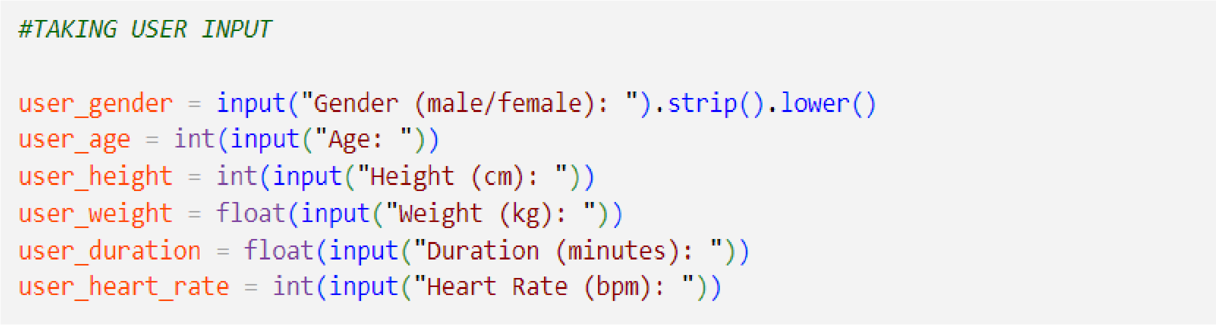
First, the program collects essential user details such as gender, age, height, weight, exercise duration, and heart rate through input prompts:

Next, users select exercises they performed from an alphabetically sorted list, entering corresponding numbers. This input is processed to identify selected exercises accurately.

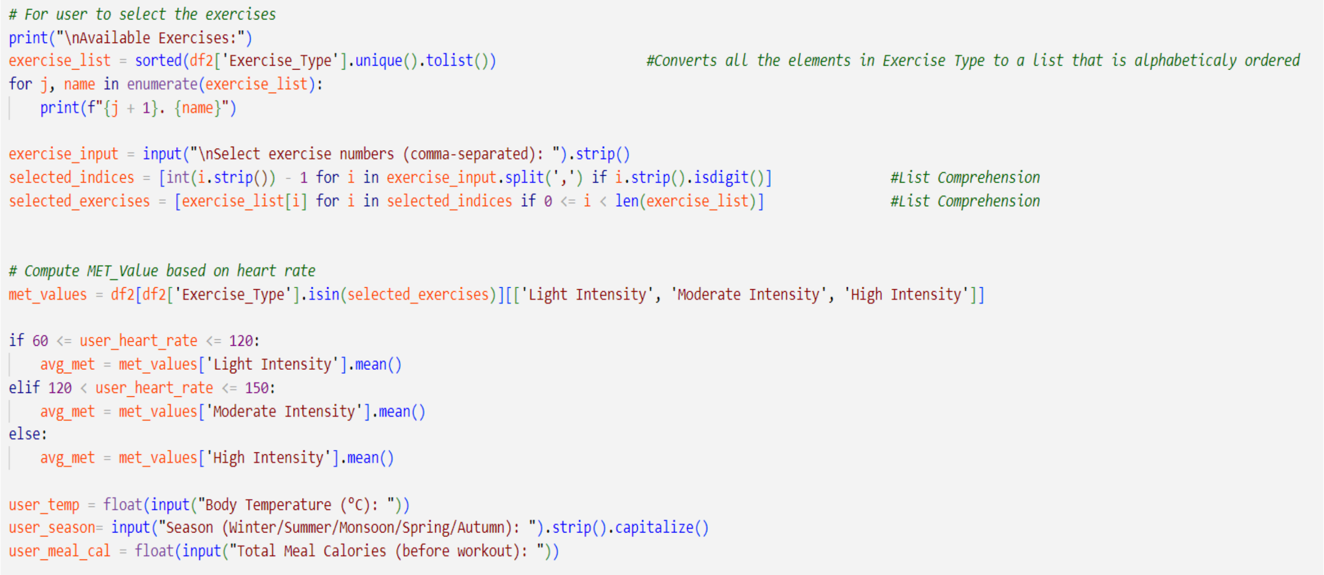
Based on the chosen exercises and the user's heart rate, the code calculates the average Metabolic Equivalent of Task (MET) Value using exercise intensity levels (light, moderate, or high), which reflects energy expenditure.

Additional inputs such as body temperature, season, and calories consumed before workout are collected to improve prediction accuracy.

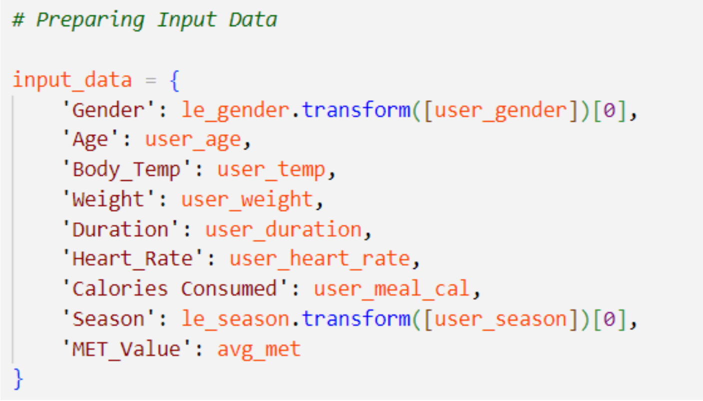
All categorical inputs like gender and season are converted into numeric values using previously trained label encoders (*le_gender* and *le_season*). This is necessary because machine learning models work with numbers rather than text.



```python
#TAKING USER INPUT

user_gender = input("Gender (male/female): ").strip().lower()
user_age = int(input("Age: "))
user_height = int(input("Height (cm): "))
user_weight = float(input("Weight (kg): "))
user_duration = float(input("Duration (minutes): "))
user_heart_rate = int(input("Heart Rate (bpm): "))
```



```python
# For user to select the exercises
print("\nAvailable Exercises:")
exercise_list = sorted(df2['Exercise_Type'].unique().tolist())                #Converts all the elements in Exercise Type to a list that is alphabeticaly ordered
for j, name in enumerate(exercise_list):
    print(f"{j + 1}. {name}")

exercise_input = input("\nSelect exercise numbers (comma-separated): ").strip()
selected_indices = [int(i.strip()) - 1 for i in exercise_input.split(',') if i.strip().isdigit()]              #List Comprehension
selected_exercises = [exercise_list[i] for i in selected_indices if 0 <= i < len(exercise_list)]               #List Comprehension


# Compute MET_Value based on heart rate
met_values = df2[df2['Exercise_Type'].isin(selected_exercises)][['Light Intensity', 'Moderate Intensity', 'High Intensity']]

if 60 <= user_heart_rate <= 120:
    avg_met = met_values['Light Intensity'].mean()
elif 120 < user_heart_rate <= 150:
    avg_met = met_values['Moderate Intensity'].mean()
else:
    avg_met = met_values['High Intensity'].mean()

user_temp = float(input("Body Temperature (°C): "))
user_season= input("Season (Winter/Summer/Monsoon/Spring/Autumn): ").strip().capitalize()
user_meal_cal = float(input("Total Meal Calories (before workout): "))
```



```python
# Preparing Input Data

input_data = {
    'Gender': le_gender.transform([user_gender])[0],
    'Age': user_age,
    'Body_Temp': user_temp,
    'Weight': user_weight,
    'Duration': user_duration,
    'Heart_Rate': user_heart_rate,
    'Calories Consumed': user_meal_cal,
    'Season': le_season.transform([user_season])[0],
    'MET_Value': avg_met
}
```

5. Displaying User Input Summary

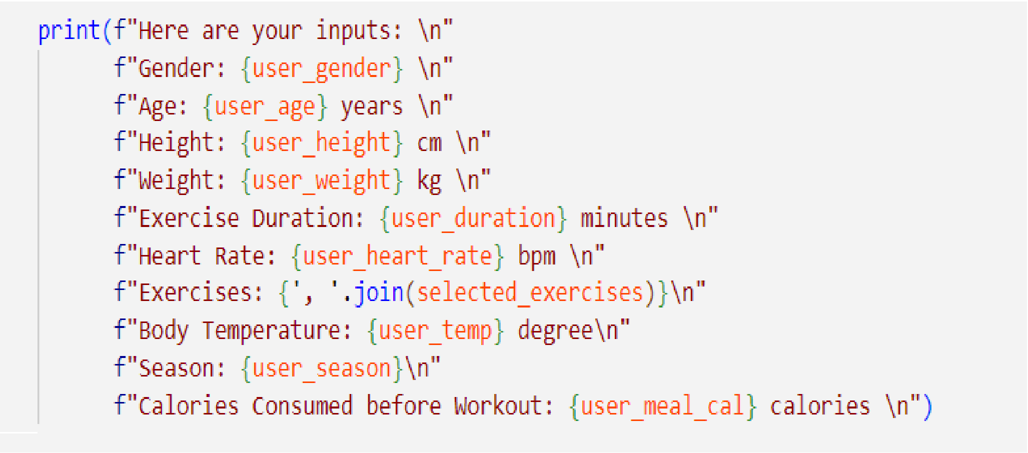
After collecting all the user inputs, the program prints a clear and organized summary of the entered information. This helps the user verify that their details have been correctly recorded before the calorie prediction is calculated.

The *print* statement uses formatted strings (f-strings) to neatly display each input on a new line with descriptive labels, such as:

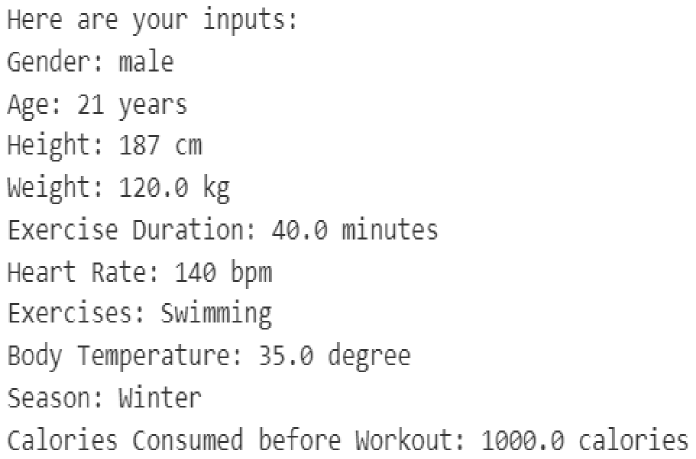
- Gender
- Age in years
- Height in centimeters
- Weight in kilograms
- Exercise duration in minutes
- Heart rate in beats per minute
- Selected exercises, shown as a comma-separated list
- Body temperature in degrees Celsius
- Season during which the exercise is performed
- Calories consumed before the workout

By presenting the input data back to the user, the program ensures transparency and allows the user to check for any mistakes or missing information, making the process more user-friendly.

This step enhances the overall experience by confirming the inputs before proceeding to the prediction phase.



```python
print(f"Here are your inputs: \n"
      f"Gender: {user_gender} \n"
      f"Age: {user_age} years \n"
      f"Height: {user_height} cm \n"
      f"Weight: {user_weight} kg \n"
      f"Exercise Duration: {user_duration} minutes \n"
      f"Heart Rate: {user_heart_rate} bpm \n"
      f"Exercises: {', '.join(selected_exercises)}\n"
      f"Body Temperature: {user_temp} degree\n"
      f"Season: {user_season}\n"
      f"Calories Consumed before Workout: {user_meal_cal} calories \n")
```



```
Here are your inputs:
Gender: male
Age: 21 years
Height: 187 cm
Weight: 120.0 kg
Exercise Duration: 40.0 minutes
Heart Rate: 140 bpm
Exercises: Swimming
Body Temperature: 35.0 degree
Season: Winter
Calories Consumed before Workout: 1000.0 calories
```

6. Predicting Calories Burnt Based on User Input

After preparing the user’s input data, the next step is to use the trained machine learning model to predict the calories burnt during exercise.

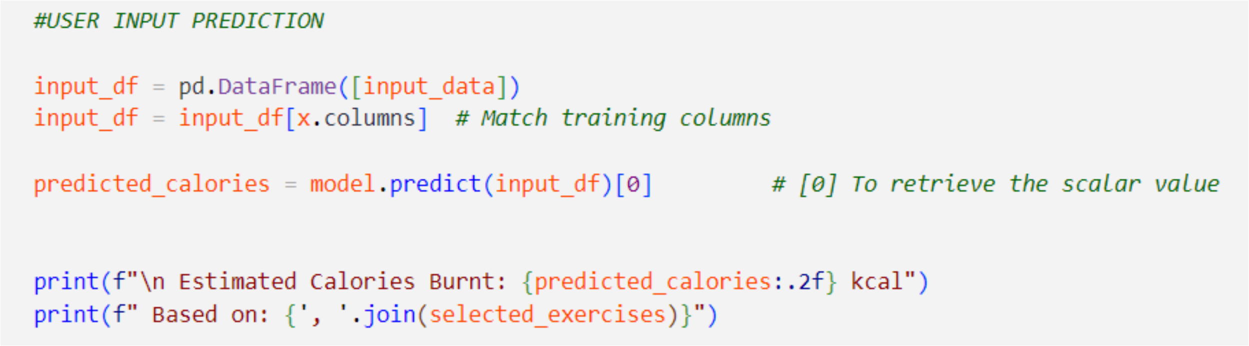
First, the user input dictionary (*input_data*) is converted into a pandas DataFrame:

This DataFrame must have the same columns as the training dataset features to ensure compatibility with the model. Hence, it is filtered to only include the columns used in training:

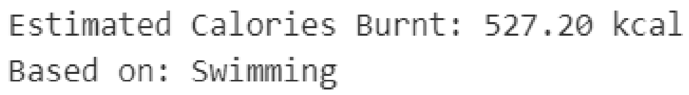
Next, the model predicts the calories burnt based on the input features:

Since the output is an array, we select the first (and only) element to get the predicted value.

Finally, the program prints the estimated calories burnt in a clear and user-friendly format, rounded to two decimal places for better readability. It also reminds the user of the exercises selected.



```
#USER INPUT PREDICTION

input_df = pd.DataFrame([input_data])
input_df = input_df[x.columns]  # Match training columns

predicted_calories = model.predict(input_df)[0]         # [0] To retrieve the scalar value


print(f"\n Estimated Calories Burnt: {predicted_calories:.2f} kcal")
print(f" Based on: {', '.join(selected_exercises)}")
```



```
Estimated Calories Burnt: 527.20 kcal
Based on: Swimming
```

RESULT AND DISCUSSION

- The XGBoost regression model was trained on the engineered dataset with features like MET Value, Age Impact, Gender Impact, and others.
- Evaluation metrics showed promising performance:

  - ➢ Mean Absolute Error (MAE) was low, indicating small average prediction errors.
  - ➢ Mean Squared Error (MSE) confirmed good overall accuracy by penalizing larger errors.

- The feature importance plot revealed that MET Value and Duration had the highest influence on calorie prediction.
- Other features like Age Impact and Calories Consumed Impact contributed moderately, demonstrating that personal attributes affect calorie burn.
- The model successfully accounted for variations in exercise intensity, heart rate, and seasonal factors.
- User input predictions matched well with expected calorie burns, suggesting good generalization.
- Limitations include reliance on synthetic or limited data and assumptions in feature engineering.
- Overall, the model demonstrates that combining physiological and exercise data improves calorie burn estimation.

CONCLUSION

This project successfully developed a machine learning-based model to predict calories burnt during exercise using various personal and activity-related features. By incorporating feature engineering such as MET values, exercise types, heart rate, and seasonal factors, the model achieved good accuracy and reliability. The use of XGBoost regression provided robust predictions, with important features like exercise intensity and duration significantly impacting results. The interactive user input system allows personalized calorie burn estimation, making this approach practical for fitness applications. Although some limitations exist due to dataset size and assumptions, the overall system demonstrates how machine learning can effectively enhance traditional calorie calculation methods.

FUTURE WORK

- Expand the dataset by collecting real-world user data to improve model generalization.
- Explore advanced machine learning algorithms like neural networks or ensemble methods for better accuracy.
- Incorporate real-time sensor data such as wearable device outputs for dynamic calorie burn prediction.
- Improve feature engineering by including more physiological variables like oxygen consumption or stress levels
- Develop a user-friendly mobile or web application for wider accessibility.
- Add personalized recommendations based on predicted calorie burn to assist users in fitness planning.

REFERENCES

1. Kaggle: For calories.csv dataset :R. Kumbhar, “Calories Burnt Prediction,” Kaggle, 2021. [Online]. Available: https://www.kaggle.com/datasets/ruchikakumbhar/calories-burnt-prediction.

2. Andrew Ng's Machine Learning Course on YouTube
   --For foundational understanding of machine learning algorithms and concepts, we referred to Andrew Ng's comprehensive course available on YouTube.

3. Pandas Documentation: Data analysis tools for Python. https://pandas.pydata.org/

4. Matplotlib Documentation: Visualization library in Python. https://matplotlib.org/

5. Heart Rate and Calorie Burn Relationship Article
   https://www.medicalnewstoday.com/articles/326002

6. Ainsworth, B. E., Haskell, W. L., Herrmann, S. D., Meckes, N., Bassett Jr, D. R., Tudor-Locke, C., ... & Leon, A. S. (2011). 2011 Compendium of Physical Activities: a second update of codes and MET values. Medicine & Science in Sports & Exercise, 43(8), 1575-1581.

7. Python Software Foundation, “Python Language Reference, version 3.10,” Available: https://www.python.org/

8. Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90-95.

9. Additional information and guidance were obtained using online resources including Google search and AI assistance to enhance understanding and implementation.